SPECIFICATION

TITLE

COMPUTER SYSTEM AND METHOD FOR SECURING MARKET PROFITS OF FINANCIAL INSTRUMENT INVESTMENTS

BACKGROUND OF THE INVENTION

FIELD OF THE INVENTION

[0001] The present invention is directed to a computer system and method for securing market profits of financial instrument investments.

DESCRIPTION OF THE RELATED ART

[0002] Profit-taking refers to the sale of financial instrument investments (the term “financial instrument” or “instrument” used here and throughout refer to stocks and/or other financial securities) that secure the market profits achieved after a rise in prices in order to convert these profits into hard currency.

[0003] It is currently standard for banks or brokers to declare a (usually time-limited) fixed stop rate. Given rising instrument rates, this must be permanently worked out by self-initiative of the bank or broker customer (an instrument investor) or on the basis of respective consultation of the bank or of the broker with the customer. When no adaptation takes place and the instrument rates again drop down to the (fixed) stop rate, the sale of the instrument is only triggered by rates hitting this stop rate, resulting in not realizing the paper profits that were achieved in the meantime.

SUMMARY OF THE INVENTION

[0004] The present invention is thus based on the object of securing market profits without requiring consultation with the customer and without complex customer self-initiative.

[0005] This object is inventively achieved by a computer system for securing market profits of financial instrument investors having:

- a mechanism for electronic storing of a yield target for a purchased species of financial instrument,

- a mechanism for electronically acquiring the current rate of the instrument species,
- a mechanism for calculating the yield and comparing the calculated yield to the yield target,
- where the mechanism for calculating and comparing is configured such that, if necessary, it implements one or more recalculations and recomparisons with rates newly acquired by the mechanism for electronically acquiring the current rate until the calculated yield is higher than the yield target,
- where the mechanism for calculating is also configured such that, given the first-time and every further occurrence of this situation, it effects storing the calculated yield target as a new yield target in the mechanism for electronically storing a yield target and, if necessary, implements one or more recalculations until the calculated yield is lower than the respective, new yield target and does not lie within a tolerance value,
- the computer system also having a mechanism for outputting an instruction to sell the instrument species when the calculated yield is lower than the (new) yield target and does not lie within a tolerance value.

[0006] The invention targets, as the consideration time span, the point in time of purchase or the first point in time of purchase. The invention also establishes a stop rate as a result of a maximum of the rate course after a yield target has been reached, minus a tolerance value.

[0007] The object of the invention is also achieved by a method for securing market profits of instrument investors using a computer, the method comprising the following steps:

a) electronically storing a yield target for a purchased instrument species;

b) electronically acquiring the current rate of the instrument species;

c) calculating the yield and comparing the calculated yield to the yield target and, when the calculated yield is lower than or of the same size as the yield target, repeating the steps beginning with step b);

[0008] otherwise

a') electronically storing the calculated yield as a new yield target;

b') electronically acquiring the rate of the instrument species;

c') calculating the yield and comparing the calculated yield to the new yield target, and

[0009] repeating Steps a') through c') when the calculated yield is higher than the new yield target,

[0010] retaining the yield target and repeating the steps beginning with Step b') when the calculated yield is lower than the new yield target but lies within a tolerance value, and

[0011] outputting an instruction to sell the instrument species when the calculated yield is lower than the new target yield and does not lie within a tolerance value.

[0012] This object is also achieved by a computer program having program code that implements the inventive method on a computer. This object is also achieved by such a computer program that is stored on a computer-readable data carrier.

[0013] The computer system can be configured to output an instruction to sell only when a predetermined time interval following the purchase has elapsed.

[0014] The mechanism for calculating the yield and comparing the calculated yield to the yield target can also be configured to calculate an annualized yield, where this annualized yield may be calculated either with respect to the purchase price of the purchased instrument species or the average annual yield for run times that are shorter than one year.

[0015] According to a preferred embodiment of the invention, the computer system may be configured to calculate an instrument index for the instruments of a customer deposit, where these calculations can provide that the index curve is steady given a purchase or sale of an instrument. This calculation for calculating the yield and comparing the calculated yield to the yield target may be configured to access the index curve as a test criterion. The prescribed minimum corresponds to a “stop loss rate” for limiting market losses.

[0016] When a speculation term is considered, then an instruction to sell (sell signal) for a profit-taking is only output when a predetermined time interval, i.e., the speculation term, has elapsed since the purchase of the appertaining instrument.

[0017] According to another, particular embodiment of the invention, the computer system is configured to check whether a prescribed minimum of the rate curve has been reached, and can output a sell signal when the minimum has been reached.

[0018] In another particular embodiment of the invention, the computer system is provided with data bank for the input and, potentially, storing the following stock attributes:

- purchase price,
- time of purchase;
- number of units;
- absolute yield target;
- annualized yield target;
- checking the yield target with reference to an instrument index;
- maximum of the rate curve after the yield target has been reached; and
- prescribed minimum of the rate curve.

[0019] These attributes can take into consideration a speculation term, where the mechanism for determining a maximum of the rate curve is provided for access onto the data bank in order to store a maximum of the rate curve.

[0020] The mechanism for doing this is a component part of a computer.

[0021] Beneficially, the invention also contemplates a server computer for offering instrument quotes and a server computer for outputting sell orders that are connectible to the computer via a network.

[0022] The computer system may also include:

- web generation utilities for providing a user interface;
- user groups storage facilities via which access rights are assigned to each user; and

- an administrator user function where the storage facilities are configured to permit the administrator to enter the users of a user group and their access rights.

[0023] The computer system may further include:

- a customer data entry system by which each customer is assigned to a pair of users of the same user group;
- a contract data entry system by which each contract is assigned to a pair of users of the same user group;
- an automatic processing status generator for the customer data and the contract data; and
- a work list generator that is based on the processing status for each of the users.

[0024] These elements are preferably a component part of a server computer.

[0025] An application service provider (ASP) server computer is especially preferred. In this case, the computer system can be used by different financial providers such as estate management companies or banks. The ASP server computer preferably has a data bank system that promotes internal communication, communication with the customer and the required administration sequences.

[0026] According to a preferred embodiment of the invention, such a data bank system offers the possibility of acquiring the master data from different companies, e.g., estate management companies and banks. Furthermore, this data bank system may serve the purpose of acquiring the appertaining employees and their respective access rights.

[0027] Preferably, at least one employee of a company has the role of a "company administrator" that, among other things, gives the appertaining employee the right to input employee data and assign access rights. The company administrator is input by an administrator of the ASP server computer.

[0028] The data bank system also preferably acquires master data of the customers as well as deposit data and the parameters for securing the system (contract data, i.e., yield target, tolerance, stop loss rate, etc. that are contractually declared between consumer and consultant) relating to individual securities. Each customer and each contract is assigned to a team of users of the appertaining county, i.e., a pair of users. This allocation the size about the competence of processing, for example, sell signals.

[0029] According to another preferred embodiment of the invention, a work list is generated for each user, where this generation is based on status information allocated to the contracts and the customers.

[0030] The data bank system thus allows an effective, computer-supported work sequence for the customer and contract administration. It is particularly advantageous that the "four eyes principle" is supported by the system, i.e., that an entry is reported by an employee and is monitored and confirmed by a higher-ranking entity. This implies that entries always have at least two statuses: reported and confirmed. When an event is only reported, then this event is in fact stored in the data bank but is not processed by the system. An event can be processed by the system as soon as it is confirmed.

[0031] When, for example, a consultant has a first customer conversation, he reports the appertaining master data of the customer. For example, fund investments are offered in this first customer conversation. This consultation will usually be followed by a request to conclude a fund investment that must yet be confirmed in a further step. The consultant now reports the master data of the customer in the system. These data contain the attribute "reported" and can thus be interrogated (for example, for sending general product information).

[0032] After the logon, these master customer data are reported to the company administrator in the system, with the request to check as well as with request of a confirmation. When the company administrator corrects the master data and/or confirms them, then this customer dataset is identified with a "confirmed" attribute in the internal data bank. From this point in time, contracts, etc., can also be assigned to this customer. The customer is listed at the company administrator with the request for confirmation until the administrator outputs the confirmation.

[0033] If the customer wishes to conclude a contract with the consultant company represented by the consultant regarding the securing of an investment, then a corresponding contract is reported in the data bank. In this work execution, the consultant can also print out the necessary forms for the customer. The contract is thus considered (as identified in the data bank) "requested". Such a request is signed by the consultant and by the customer and only takes effect on the basis of a third signature of the company administrator. In addition to the signature, the company administrator must likewise confirm the requested contract in the system. From this point in time, the confirmed contract is monitored in the data bank.

[0034] When a contract is noted by the customer, then the account manager or consultant will report this noting in the system. The customer's noting then appears together with the request for confirmation at the associated consultant, who confirms the noting. A form letter can be printed out in the process of the log on by the account manager, and the letter can be presented to the consultant for signature. The corresponding contract is deleted from the data bank with confirmation of the customer's noting.

[0035] When a sell signal is generated by the computer system, i.e., the fund shares are marked to be sold, then this is reported to the account manager by the system. The account manager or consultant prints out a corresponding form for notifying the customer a well as the existing sales request (digitally stored) and submits this to the customer consultant for signature. The corresponding sell order is reported by the account manager. This application is confirmed by the consultant and the sale is initiated.

[0036] Due to the deletion of an employee (consultant), associated contracts (i.e., the contracts between final consumer and consultant that contain the securing parameters for an investment) must be assigned to a different employee. To achieve this, the deletion of the employee is suppressed until every individual contract has been assigned to a different consultant.

[0037] In general, contracts and customers are handled by teams. Teams are pairs of employees of a firm that are composed of a consultant and an account manager. These teams are assigned to every contract and customer. Every change in status of a customer or of a contract is co-logged by a change of the attributes of the corresponding object. This permits replication at any time of who elicited specific status changes.

[0038] As a result of the ASP approach, the connected companies or consultants require only a slight infrastructure (e.g., a computer, browser, and/or Internet connection) in order to be able to work with this computer system. Moreover, a plurality of employees (users of the computer system) can work in parallel on the same customer and contract inventory individually or in the team. This assures that different members of a team can efficiently implement the tasks given a multitude of contract applications or contract notings.

[0039] This also assures, due to the imaging of the work flow of the consultant, that the concluded contracts with the customer are represented as closely as possible by the data bank contents.

[0040] The selected ASP approach is particularly advantageous since the connected consultants or companies need comprise only a slight infrastructure in order to be able to be connected to the system. Furthermore, the participants in this system can log on from any arbitrary location (presuming a terminal and Internet connection). This additionally permits giving the final consumer of the connected consultants access to the current deposit values and yields achieved as well.

[0041] The following aspects are especially advantageous:

- The various combinations of the contracts allow a greater flexibility in the investment of the customer assets. Furthermore, these security parameters are not limited in time as is generally standard.
- The possibility of forming teams of an account manager and consultant corresponds to the standard procedure in larger consulting companies. As a result of this system functionality, the work can be divided among the team members.
- By introducing access rights, it becomes possible to transfer specific responsibilities that, for example, exist in a consulting company directly into the system.
- The advantages of the four-eyes principle (i.e., the necessity of confirmation by the company administrator given business-relevant events) is that the company administration or company management is always informed about changes in the customer or contract inventory and

can also consult here with the participating consultants. This functionality of the system achieves a great transparency regarding the customer set but also regarding the contracts that have been concluded as well as their progress.

- The imaging of the work flow given consultations in the system permits the respective relationships or contract statuses with the customer to be imaged as closely as possible. Thus, for example, when a contract is selected for the customer, a printout of the necessary forms may be simultaneously prepared. As soon as a contract is signed by the customer, this is reported in the system and presented to the company management for signature. The company management/company administration should confirm this contract parallel to the signature. The contract can thus only be activated when this has been enabled or signed by the company administration.

[0042] Expediently, the invention may provide a server computer for offering instrument rates and a server computer for outputting sales contracts, such server computers being connectible to the ASP server computer via a network.

[0043] Advantageously, the invention may provide at least one computer of a bank or the like that is connectible to the ASP serve computer via a network.

[0044] Finally, the ASP server computer can be designed such that at least one customer of the bank or the like can communicate directly via the network with the ASP server computer.

[0045] An inventive method is also described below.

[0046] The invention is based on the nonobvious realization that, due to the specific rate monitoring and interpretation, consultation with the instrument investor or, respectively, a complex self-initiative of the investor are not required. An interaction between the customer and the customer's bank or broker is thus superfluous, as is an external data bank access. On the contrary, the adaptation is fully automated using the inventive computer system without an requiring an external data bank access and the setup of a data transmission.

[0047] The indication to sell (sell signal) can be electronically output, for example, on a picture screen, so that the bank consultant or broker and/or the customer himself can take note of the sell recommendation. The electronic output of the sell signal can take place, for example, in the form of an electronic message (e.g., an e-mail or work list). Alternatively or additionally, the sale can be fully automatically sequenced on the basis of the sell signal in that, for example, a corresponding sell order is generated by the computer system and is communicated via a dataline to the appertaining fund corporation or to an on-line broker.

DESCRIPTION OF THE DRAWINGS

[0048] For the purposes of promoting an understanding of the principles of the invention, reference will now be made to the preferred embodiments illustrated in the drawings, and specific language will be used to describe these embodiments. No limitation of the scope of the invention is intended by this specific language, and the invention should be construed to encompass all embodiments that would normally occur to one of ordinary skill in the art.

Figure 1 is a block diagram showing a first embodiment of an inventive computer system for commerce with securities;

Figure 2 is an exemplary data structure diagram of the customer data bank for the computer system of Figure 1;

Figure 3 is a graph illustrating an embodiment of the inventive method on the basis of a rate course; and

Figure 4 is a block diagram showing an embodiment of an inventive computer system for commerce with securities having an ASP server computer.

DETAILED DESCRIPTION OF THE INVENTION

[0049] Figure 1 shows a computer system with a computer 1 having a rate databank 2 and a customer databank 3. The rate databank serves for storing current security rates that are required for the evaluation.

[0050] The master data of a customer as well as one or more security identifiers of the customer deposit are stored in the customer databank 3. The customer can specify for a specific security whether to monitor an absolute yield target or an annualized yield target, namely with reference to the rate course of the appertaining security or with reference to a customer deposit index.

[0051] Additional entries into the customer databank 3 can include whether the customer wishes a profit securing stop rate for securing market profits and how great the potential tolerance value is. Furthermore, a stop loss rate as well as the observation of a speculation term can also be optionally input. A preferred embodiment of the customer data bank 3 is explained in greater detail below with reference to Fig. 2.

[0052] The input of the data into the customer data bank 3 may ensue via a user interface 4 that, for example, is made available by the operating system of the computer 1 and enables inputs as well as outputs via monitors 5 connected to the computer 1.

[0053] The computer 1 may also have an evaluation program 6 with a program module 7 for checking whether a yield target has been achieved, a program module 8 for calculating a customer-specific security index, a program module 9 for determining the profit securing stop rate, a program module 10 for monitoring when the stop loss rate is reached, and a program module 11 for generating a work list that, for example, indicates sales to be undertaken.

[0054] The computer 1 may be connected via a network 12 (e.g., the Internet or telephone network) to a server computer 13 from which current security prices can be fetched. Furthermore, the computer 1 may be connected via the network 12 to a server computer 14 of a fund corporation or an online broker in order to activate cell orders. Alternatively, for a server computer, for example, a fax machine can also be present at the side of the fund corporation in order to be able to accept sell orders communicated via fax.

[0055] During operation of the computer 1, the current security prices of the securities covered in the customer data bank 3 are interrogated by the server computer 13 via the network 12 and are stored in the rate data bank 2. The evaluation program 6 then accesses the rate data bank 2 in order to evaluate the rate curve according to the customer prescriptions stored in the customer data bank 3.

[0056] The program module 7 then automatically checks whether the desired yield target has been achieved. When the customer has prescribed the achievement of an absolute yield target, the program module 15 is started; when, in contrast, an annualized yield target has been prescribed by the customer, the program module 16 is started. It is also a determining factor for the type of calculation of the yield as to whether the customer wishes the yield calculation related to an individual stock or related to the index course of his customer deposit.

[0057] The program module 15 is started, i.e., when a check is carried out to see whether the rate course or the index course has reached a predetermined value.

[0058] It is advantageous, precisely for investments in investment funds, to specify annualized yield targets or to specify these with interest profits. This permits long-term yield targets to be tracked. The basic idea is to equip every contract type with yield target with an annualized or interest-related yield.

[0059] Since, in particular, the formula for the interest-related yield is unstable for short terms (i.e., small changes in performance generate great changes in the interest-related yield), the yield is either calculated with respect to purchase price or the average annual yield for run times that are shorter than one year. The following equation is based on day intervals but can also be modified for week intervals, etc., and is defined with the average annual yield for time spans shorter than one year:

[0060] Let (s_t) be the result of all prices for a stock s at points in times in day t. Then let

[0061] $R(s,t_0,t_1)=\frac{S_{t_1}-S_{t_0}}{S_{t_0}}$ be the yield with respect to the purchase price s_{t_0} at time t_0 as well as the current time t_1 with $t_0 < t_1$.

[0062] The annualized yield is then calculated by

[0063] $$R_A(s,t_0,t_1)=\begin{cases}\frac{t_1-t_0}{T}R(s,t_0,t_1), t_1-t_0 \le T \\ (1+R(s,t_0,t_1))\frac{T}{t_1-t_0}-1, t_1-t_0 > T\end{cases},$$

[0064] where T is usually set to 365 and T = 366 only applies given leap years.

[0065] The Equation $\frac{t_1 - t_0 T}{T} R(s,t_0,t_1), t_1 - t_0 \leq T$

[0066] corresponds to an investment behavior that would sell the investment after the time period $[t_0, t_1]$ and would forego a re-investment.

[0067] This calculating method is advantageous in order to also be able to evaluate short-term profits of security investments (investment period < one year) over the term of a year as well as to enable a comparison with risk-free investments with re-investment (for example, capital, etc.).

[0068] When the check of an annualized yield target is requested by the customer, the program module 16 is thus started that calculates the annualized yield, as indicated above, and compares this to the yield target.

[0069] In general, investments precisely in the fund field can be "continuously" implemented. For example, shares of a fund can be acquired on a monthly basis with respect to a fixed investment sum, resulting in the investor acquiring many shares when the fund has a low price and fewer shares when the price is high. In general, the investor thus levels out price fluctuations (this effect is generally called the "cost-average effect") and thus reduces investment risks. Equipping this investment method with the above-recited securing methods as well as the calculation of yield targets (with respect to purchase price or "annualized") also makes it possible to secure this form of investment.

[0070] The basic idea is that the calculation of the yield of the accumulated investments at every purchase is time dependent on the basic investment. Purchase times can be flexibly handled and can be indicated at every possible time. In general, however, it is standard to select equidistant, monthly points in time given investments in funds (fund saving).

[0071] Every investment point-in-time is associated with a purchase sum as well as the number of shares.

[0072] It is advantageous that continuous investments can be monitored and secured with the above-described methods.

[0073] A general version for evaluating the performance of deposits is provided below that can be adapted to the requirements of fund-saving systems (regular purchase of securities). This is analogous to the calculation of index values such as the German stock index of the Deutsche Börse AG:

[0074] Let $s_k(t)$ be the price of a security of a deposit position k at time t. Let the times again be given in daily resolution and let the price $s_k(t)$ be the closing quotation of the stock (other time resolutions are possible). Let $n_k(t)$ be the number of stocks of the deposit position k at time t. The deposit value at time t is then calculated by

[0075] $D(t) = \sum_k s_k(t) n_k(t).$

[0076] Problems in the calculation of the performance of the deposit are possible purchases or sales of deposit positions at specific points in time. Let these times at which the deposit changes be defined by $t_1, \ldots . t_N$. Let

[0077] $D^*(t) = \sum_k s_k(t) n_k(t) + S_M n_M$ also be the modified deposit value at a purchase time by the addition of a further deposit position M at n_M units and a purchase price S_M.

[0078] Then let

[0079] $i^*(t) = \max\{\ i \mid i\ \in \{1,...,N\}\ \wedge t_i \le t\ \}$ be the index of the most recent purchase time t_i and let $K_0, \ldots, K_N$ be chaining factors that can be recursively calculated by

[0080]

[0081] $K_1 = \frac{D(t_i - 1)}{D^*(t_{i)}} K_{i-1}$ with $K_0 = 1$.

[0082] A chained index time series can then be calculated by $I(t) = K_{i^*(t)} D(t) I$.

[0083] The index time series I(t) must thereby be implicitly calculated via the recursively defined chaining factors K_i with D(t) = 1 for $t < t_1$, where I can assume an arbitrary value (for example, 1000 at the DAX) but should assume the most recent available overall purchase volume of all deposit positions in the calculation of the yield of the deposit.

[0084] When an entry is found in the customer data bank 3 for the appertaining security by which the yield target is to be checked on the basis of the deposit-specific index, the program module 8 is started in order to calculate the index I(t) according to the above-cited equation.

[0085] When the yield target has been achieved, the program module 7 starts the program module 9 for determining the yield-securing stop rate. When the yield target is reached for the first time, the appertaining price minus the tolerance is stored as yield-securing stop rate being stored in a memory of the computer 1 by the program module 9.

[0086] The corresponding security or customer deposit is subsequently located in the monitoring for securing the achieved market profits. During the course of further commerce, this means that security prices are monitored by the program module 9 to see whether these, minus the tolerance, lie above or below the previously stored profit-securing stop rate. When the current price or index value, minus the tolerance, lies above the previously stored profit-securing stop rate, then the current market price or index value, minus the tolerance, is deposited as a new profit-securing stop rate instead of the preceding one, and is deposited in the memory of the computer 1.

[0087] When, in contrast, the current market price or the current index value downwardly exceeds the preceding price or index value, then a check is carried out to see whether this downward transgression lies within the range of tolerance specified by the customer and stored in the customer data bank 3 (i.e., above the stop rate). When the downward transgression lies within the range of tolerance, then the previous profit-securing stop rate is retained; no action over and above this takes place.

[0088] When, in contrast, the tolerance range (i.e., the scope of fluctuation) is downwardly exceeded (i.e., the price or index value drops below the stop rate) then the program module 9 automatically generates a sell signal.

[0089] When the customer has indicated a stop loss rate, then the program module 10 checks for every new security price whether this stop loss rate has been reached or downwardly exceeded. When the stop loss rate is reached or downwardly exceeded, the program module 10 automatically outputs a sell signal.

[0090] The sell signal is generated by the program modules 9 and 10 can be output as automatic sell orders, these being output by the computer 1 via the network 12 to the server computer 14 or the corresponding fax machine, so that the sale of the appertaining security or securities is automatically carried out. Alternatively, the program module 11 is started that generates a work list that contains the sales to be made. This work list is displayed on the monitor 5 so that a user can process it.

[0091] The computer 1 can, for example, be utilized for commerce with stock funds. Given such stock funds, the take-back price is fixed on every trading day, for example at 2:00 p.m. or at 4:00 p.m. Fund shares can then be returned to the fund corporation at these daily quotations on the succeeding trading day. For this application, the server computer 13 calls the prices of the stock funds fixed by the market daily and these are evaluated by the evaluation program 6. Subsequently, a work list is generated with the program module 11 which is then capable of being processed by a user in that the user, for example, sends corresponding sell orders by telefax or online to the appertaining fund corporation or the appertaining online broker.

[0092] Preferably, the evaluation program 6 is configured such that it is started automatically at a specific point in time, for example, after the end of a trading day, in order to evaluate the daily quotations and generate a corresponding work list. The user of the computer 1 can then process this on the next morning.

[0093] Alternatively or additionally, the computer 1 can also be configured such that the commercial course of specific stocks and securities is tracked constantly in that the appertaining security prices are loaded by the server computer 13 as soon as they are available. The evaluation program 6 is then also started correspondingly frequently. For this application, which is time-critical due to the constant fluctuation of the prices, the direct electronic forwarding of a sell signal output by the program modules 9 or 10 to the server computer 14 is recommended.

[0094] Figure 2 shows an embodiment of the customer data bank 3 of Figure 1. One or more contracts are present for each customer in the customer data bank, these contracts being identified by the security identifier of the appertaining security, i.e., for example, of the stock or investment fund. The customer data bank contains the purchase price as well as the purchased time and the number of units of the purchased stocks or fund shares for every contract.

[0095] When the customer has indicated an absolute yield target, the data bank contains this yield target in percentages or as an absolute value referred to the price of the appertaining security or referred to an index value.

[0096] When the customer desires an annualized yield target, then this is likewise noted in the data bank. When the absolute yield target or the annualized yield target is to be checked with reference to the index course of the customer deposit, a corresponding entry is present in the data bank. Furthermore, the data bank contains a field for the profit-securing stop rate that is determined by the program module 9 (see Figure 1). As warranted, the tolerance value requested by the customer is also input into this field.

[0097] The data bank also contains a field for inputting a stop-loss rate as well as a field for specifying whether a speculation term should be observed or not.

[0098] Figure 3 illustrates the operation of one embodiment of the inventive method based pm the rate course of a specific security. The time is entered toward the right and the current price for the security is entered toward the top. At a time t_0, a specific unit number of the security is bought for a customer at a price of, for example, 100 euros via an investment consulting company.

[0099] This purchase is based on a corresponding contract with the customer which declares that an absolute yield target according to the rate course should be evaluated, and that market profits should be secured with a profit-securing stop rate. A stop-loss rate has not been declared, nor has observing a speculation term. The customer data bank 3 (see Figure 1 and Figure 2) serves the purpose of storing the appertaining contract parameters that have been input via the user interface 4 of the computer 1 (see Figure 1).

[00100] A price of 130 euros has been declared with the customer as an absolute yield target. A fluctuation tolerance of 5% of the absolute yield target has been declared as a tolerance value for the deviation from the maximum of the rate course.

[00101] After the purchase at time t_0, the price initially drops, for example, until it rises again. At time t_1, the price has reached the absolute yield target in the

amount of 130 euros. This is identified by the program module 15 and the program module 9 is started (see Figure 1).

[00102] The price maximum of 130 euros that has been reached, minus the tolerance of 5% (6.5 euros), is then stored as profit-securing stop rate, i.e., a stop rate of 123.5 euros is entered in the data bank. The price continues to rise during the following time, so that the program module 9 continuously stores no profit-securing stop rates that respectively reflect the maximum quotation minus the tolerance.

[00103] At time t_2, the price reaches its initially highest value of 150 euros. This value minus the tolerance of 5% of this value, i.e. 142.5 euros, is then the current profit-securing stop rate at this time t_2.

[00104] Subsequently, the price drops again and reaches a value of 145 euros at t_3. Due to the dropping rate course, the profit-securing stop rate of 142.5 euros is maintained. The program module 9 (see Figure 1) then checks whether the rate course is within the tolerance value, i.e., the fluctuation tolerance of 5% (above 142.5 euros). Since the change only amounts to 5 euros here, the fluctuation lies within the tolerance value so that no sell signal is generated.

[00105] After time t_3, the price again rises steadily until time t_4. Accordingly, the profit-securing stop rate is repeatedly adapted to the current price. After time t_4, the price of 200 euros drops to below 190 euros at time t_5. Since the last new profit-securing stop rate lies at 190 euros (200 euros minus 5% tolerance), the fluctuation tolerance to the amount of 5% has now been exceeded, so that a sell signal is generated by the system.

[00106] When the customer desires adherence to a speculation term, the time between the point in time of the purchase t_0 and the sale t_5 is additionally checked to see if it is longer than the speculation; the sell signal is generated when this is the case.

[00107] When the customer requests adherence to a stop loss rate of, for example, 80 euros, then this rate is entered in the customer data bank 3 and is constantly monitored by the program module 10 (see Figure 1). In the exemplary case of Figure 3 under consideration, this would result in a sell signal being

generated immediately after the purchase, since the price drops below the stop loss rate, namely to 70 euros.

[00108] Figure 4 shows another embodiment of an inventive computer system that uses an ASP.

[00109] The computer system contains an ASP server computer 17 that - similar to the embodiment of Figure1 - is connectable to server computers 13 and 14 via a network 12 for querying security prices or for outputting sell orders.

[00110] The ASP server computer 17 contains a web front end 18 for generating web pages in order to make a user interface available. Furthermore, the ASP server computer 17 contains middleware 19, i.e., software components that are addressed via the user interface made available by the web front end 18. The ASP server computer 17 further contains a back end 20 having a data bank system. The data bank system, for example an SQL data bank, stores status information.

[00111] The data bank structures are encapsulated by the objects of the middleware, i.e., every user action is implemented on the middleware (the objects) and checked for consistency and errors. As soon as a status change is accepted by the middleware, this status is stored in the data bank. Statuses can thus be transferred from web page to web page, since, differing from classic programs, each generation of a web page generates a new server process that has no information about the "history" of the user interactions per se.

[00112] The workflow should furthermore be closely adapted to the workflow of the asset consultant so that actions that the consultant implements and declares with the customer can be imaged as closely as possible in the data bank. To this end, it is necessary to model all supported units as data bank object and encapsulate them with a corresponding object in the middleware.

[00113] For example, this modeling allows the definition of teams (account manager and consultant) who together handle contracts as well as customers in the system. Due to this flexibility, parameters are allocated to each "object", i.e., each contract or customer, that later allow replication of who modified which object. Moreover, specific information can be allowed with specific units, for example, the members of a team who handle a specific customer, on the basis of these auxiliary parameters, but can be blocked for others.

[00114] The basic object modeling is described below by way of example.

[00115] In the ASP solution, various companies that are represented by data bank objects can be integrated. Employees are assigned to each company. An access right is allocated to each employee. This access right is either in ASP administrator, company administrator, consultant or account manager.

ASP-ADMINISTRATOR

[00116] The ASP administrator is responsible for the operation of the system and is generally allowed to monitor every process of a licensed customer as well as to intervene in statuses. Thus, an ASP administrator is allowed to:

- add, delete or modify companies;
- add, delete and modify employees of companies;
- produce company statistics with respect to the customers and the concluded contracts;
- have concluded contracts displayed;
- view customer data, etc.

COMPANY ADMINISTRATOR

[00117] The company administrator is responsible for all work sequences within a company and also serves as a monitoring entity with respect to the customer relationships: for example, the company administrator is allowed to

- add, delete employees of this company or modify their master data;
- view all customer contracts of his company; and
- view statistics about employees (and their customers) of his company.

[00118] The company administrator must counter-sign requested monitoring contracts as well as confirm logons of customers of his company. The business manager of a consulting company is usually entered as company administrator.

CONSULTANT

[00119] The consultant has direct customer contact and offers profit-securing contracts that are later inserted into the system to his (final) consumer. The consultant is allowed to:

- log customers on for acceptance into the system;
- request the deletion of customers; as well as
- confirm contract applications as well as contract noticings.

[00120] In general, the consultant only has access to his own customers.

ACCOUNT MANAGER

[00121] The account manager is linked to a consultant and prepares contracts of the final consumer in general that were communicated by the consultant and that must be confirmed in the system by the consultant.

TEAM FORMATION

[00122] Account managers and consultants can be joined as teams by the company administration. All information that relates to a part of the team are made available to the entire team.

[00123] Such an ASP solution allows the administration of multi-level access hierarchies and thus makes it possible that a plurality of companies having an arbitrary plurality of customers and employees as well as contracts can be administered at a central entity. Additionally, a workflow is defined that allocates clear responsibility of the users when dealing with the system.

[00124] This assures that all sell signals can be taken into consideration and that concluded contracts agree with the legal situation or derived responsibility situation that are usually (but need not be) administered outside the system. The comprehensive storing of all necessary parameters also assures that all investments that are placed in the system can be subjected to a comprehensive evaluation. Included here are evaluations that refer to various hierarchies (company, employee, final consumer).

[00125] In the embodiment of Figure 4, the web front end 18 is supported on HTML as well as by dynamic web pages generated by the server (CGI or other programs). This corresponds to the classic client-server model in which a client (in this case, the consultant) triggers actions on the server via a respective user interface. In this system, the client need only have a computer with an Internet connection as well as a browser (HTML). The actions on the server are triggered by the user by calling web pages (via URL). The entire program execution is thus implemented on the server. The server may be composed of five different modules: company unit, employee unit, customer unit and the contract unit as well as a scanner (background process) that monitors the input contracts in view of potential sell signals.

[00126] An arbitrary plurality of companies can be administered in the company unit in the system. Each company contains employees (the client users) that are assigned to this company. In general, these employees can only access data of their allocated company. In turn, customers are allocated to the employees. Employees differ on the basis of four different, possible access rights (see above). On the basis of these assigned rights, the server decides what program flow and, thus, which information should be made available to the client (the employee).

[00127] The company administration can form pairs of employees (consultant and account manager) that respectively handle the customer as well as contracts together. Customers may be allocated to each employee from the customer unit. Each employee can only view or modify the customers or contracts assigned to him. Deposits may be allocated to each customer, these in turn containing the contracts (i.e. fund investments plus securing strategies) communicated from the consultant.

[00128] The various units may thus be strictly hierarchically oriented.

[00129] By specifying a company, a user name and a password, a client (employee) can log on at the system. The server can undertake an unambiguous identification on the basis of the company and employee unit. Moreover, the work environment may be adapted on the basis of the priority (access rights) of the employee.

[00130] Since, in particular, the customer data and the contract entries in the respective modules may be provided with status information, a work list can be

produced by the server after the log on of the employee. Account managers and consultants can be presented with customer entries as well as contract entries that have not yet been confirmed (that are yet to be confirmed) as well as with current contracts accepted for monitoring by the scanner unit as well as sell positions that have been signaled.

[00131] In contrast to this, company administrators may also be presented with customer or contract applications to be confirmed and customer deletions or contract nullifications. In addition, the company administrator, by passing the four-eye principle, receives messages of his employees, i.e., whatever contract or customer applications are confirmed by the consultant himself, the system sends a message to the company administrator.

[00132] In the exemplary case of Figure 4, the back end 20 contains a company unit 21 for storing the master data of various company customers of the ASP. The companies typically comprise asset and investment consulting corporations.

[00133] The back end 20 also contains an employee unit 22 for storing master data of employees as well as access data such as user name and password as well as a pointer to the access rights unit 23 for defining the role of the appertaining employee. Every employee covered in the employee unit 22 is assigned to a specific company of the company unit 21.

[00134] The ASP administrator can add a company to the company unit 21 as well as input a company administrator for this company as an employee in the employee unit 22. A pointer to the access rights as company administrator may be generated for the appertaining employee of the company. The company administrator can then employees of his company to the employee unit 22 and assign access rights.

[00135] The back end 20 also has a customer unit 24. Each customer entry, in addition to containing the master data, contains the company affiliation, the allocation to a team (groupings of consultant and manager) of the team unit 25, the status information as well as the employee who prepared the customer dataset and that employee who most recently modified the customer dataset. A customer dataset can generally be called only by employees that belong to the indicated team

or have administrator authorization. This means that the customers within a company are separated among the various consultants.

[00136] The status information indicates what should occur with the customer dataset in the workflow. In an exemplary embodiment, REGISTERED marks the dataset as input, and signals the system that this dataset should be displayed to the company administrator for correction or confirmation. When the company administrator corrects or confirms this dataset, the status information is raised to CONFIRMED. Contracts (fund investments) can only be assigned to customers when these have the status information CONFIRMED or CHANGED. This status information is set to CHANGED when the customer dataset was manipulated. When a consultant or account manager logs customers on for deletion, the status information is set to CHECKOUT. No further contracts, etc., can now be assigned to the customer. This requested customer deletion is in turn presented to the company administrator who can confirm this (CHECKOUTCONFIRMED).

[00137] The back end 20 also contains a contract unit 26. As described above, various contract modalities are supported by the system. The contract unit contains various parameters that describe the respective contract versions in specific combinations.

[00138] All contracts are assigned to a deposit and contain a securities identifier that refers to the security to be purchased. Furthermore, a company as well as a team of this company are allocated to the contract, these being notified by the system when the contract is noticed. Analogous to customers, contracts are equipped with a status information as well as with a reference to the status-modifying employee. When a contract is logged on in the system by an employee (i.e., presented in parallel to the business management for signature), then the contract is correspondingly marked with REGISTERED. Contracts marked as REGISTERED are presented to company administrators by the system for confirmation. As a result of the confirmation of the company administrator, the contract is identified as CONFIRMED.

[00139] Only contracts that have a higher priority than REGISTERED are monitored. These contracts to be monitored are checked at certain points in time by the monitoring module dependent on the type of contract. When, for example,

contracts reach their yield target, then the contract is equipped with a stop rate and is also identified with the status information ACTIVATED. Contracts that have been identified with ACTIVATED are signaled to the respective team as being activated, so that the account manager or the consultant can send a message to the customer (automatic sending of a notification is not precluded). When a team member confirms this presentation of the system, then the status MONITORED is set. Profit securing can only begin when the status of the contract is ACTIVATED or MONITORED.

[00140] When a check shows that the contract should be set to sell because, for example, a profit-securing stop rate has been downwardly exceeded (profit securing), then the system marks this contract with TERMINATED. Contracts having this status information are again displayed to the team members. As soon as the fund shares have been sold, the team members must confirm this notification, so that the contract is marked as CHECKOUT and is presented to the company administration for confirmation.

[00141] A contract of the contract unit 26 can be linked with a deposit of the deposit unit 27 as well as with the beneficiary unit 28 and the fee unit 29. The beneficiary unit 28 serves the purpose of storing the persons benefited by the contract; the fee unit 29 serves the purpose of debiting the fees charged by a company, for example, for activating a sale.

[00142] The back end unit 20 also contains a fund company unit 30 for storing the master data of fund companies by which sales are to be activated and also contains a rate data unit 31 for storing current rate data that have been communicated from the server computer 13.

[00143] The notification unit 32 of the back end 20 serves the purpose of generating internal messages when, for example, an employee has himself activated a contract bypassing the four-eyes principle.

[00144] The customers 46 of the company - just the like the companies via their computers 45 - can communicate with the ASP server computer 17 via a network 33, for example, the Internet. Furthermore, the ASP server computer 17 can be configured such that the customers of the appertaining companies can also directly

communicate with the ASP server computer 17 via the network 33 in order, for example, to query the processing status, current price data, and sell signals.

[00145] Among other things, the evaluation program 34 accesses the contract unit 26 and has a functionality analogous to the evaluation program 6 of the embodiment of Figure 1.

[00146] The present invention has been described in terms of functional block components and various processing steps. Such functional blocks may be realized by any number of hardware and/or software components configured to perform the specified functions. For example, the present invention may employ various combinations of hardware or software elements, e.g., memory elements, processing elements, logic elements, look-up tables, and the like, which may carry out a variety of functions under the control of one or more microprocessors or other control devices. Similarly, where the elements of the present invention are implemented using software programming or software elements the invention may be implemented with any programming or scripting language such as C, C++, Java, assembler, or the like, with the various algorithms being implemented with any combination of data structures, objects, processes, routines or other programming elements. Furthermore, the present invention could employ any number of conventional techniques for electronics configuration, control, data processing and the like.

[00147] The particular implementations shown and described herein are illustrative examples of the invention and are not intended to otherwise limit the scope of the invention in any way. For the sake of brevity, conventional electronics, software development and other functional aspects of the systems (and components of the individual operating components of the systems) may not be described in detail. Furthermore, the connecting lines, or connectors shown in the various figures presented are intended to represent exemplary functional relationships and/or physical or logical couplings between the various elements. It should be noted that many alternative or additional functional relationships, physical connections or logical connections may be present in a practical sensor device. Moreover, no item or component is essential to the practice of the invention unless the element is specifically described as "essential" or "critical". Use of the word “mechanism” is not

intended as being limited to defining physical structure or entity, but can also be interpreted as a software implementation and/or methods.

10057637.021302

APPENDIX
DATA BANK DESCRIPTION OF THE VARIOUS UNITS OF THE BACK END 20

[00148] This appendix provides an exemplary listing of various back-end units.

COMPANY UNIT

[00149]

```
DROP         SEQUENCE          seq__dapt__company;
DROP         TABLE             dapt__company;
CREATE            TABLE              dapt__company

             id                               INT4   PRIMARY KEY,
             name                             TEXT   NOT NULL,
             street1                                TEXT,  NOT NULL,
             street2                                TEXT,
             city                             TEXT   NOT NULL
             state                            TEXT,
             zip                              INT4   NOT NULL
             country                          TEXT   NOT NULL
             phone                            TEXT   NOT NULL
             fax                              TEXT,
             email                            TEXT,
             www                              TEXT,
             resources                        TEXT,
             contractSequence                 INT4 DEFAULT(0),

             UNIQUE (name, street1, zip, city)
);
CREATE             SEQUENCE          seq__dapt__company;
```

[00150] Company addresses can be stored in the data bank. These company addresses can only be viewed, modified, compiled or deleted by the ASP administrator.

ACCESS RIGHTS UNIT

[00151]

```
DROP    TABLE dapt_permission;
CREATE             TABLE dapt_permission
(
             id              INT4   PRIMARY KEY,
             remark                 TEXT  NOT NULL
);

INSERT INTO dapt_permission (id,remark) VALUES (0,'ASP Administrator');
INSERT INTO dapt_permission (id,remark) VALUES (1,'Company Administrator');
INSERT INTO dapt_permission (id,remark) VALUES (2,'Customer Advisor');
INSERT INTO dapt_permission (id,remark) VALUES (3, 'Customer User');
```

EMPLOYEE UNIT

[00152]

```
DROP       SEQUENCE         seq_dapt_employee;
DROP       TABLE            dapt_employee;
CREATE     TABLE dapt_employee
(

        id                                    INT4   PRIMARY KEY,

        firstName          TEXT  NOT NULL,
        lastName           TEXT  NOT NULL,
        title              TEXT  ,
        gender_id          INT4  REFERENCES dapt_gender(ID) NOT NULL,
        dateOfBirth        DATE  NOT NULL,

        company_id         INT4  REFERENCES dapt_company(id) NOT NULL,
        ident              TEXT  ,
        phone              TEXT  NOT NULL,
        fax                      TEXT  ,
        email              TEXT  ,
        www                      TEXT  ,
        username           TEXT  NOT NULL,
        password           TEXT  NOT NULL,
        permission _id      INT4  REFERENCES dapt_permission NOT NULL,
        create             DATE  DEFAULT CURRENT_DATE,
        lastLogin          INT4   ,
        sessionKey         TEXT  ,

        substitute_id INT4  DEFAULT(-1),
        jobTitle      TEXT,

        UNIQUE( company_id, username)

);
CREATE     SEQUENCE         seq_dapt_employee;
```

[00153] Each employee is allocated to exactly one company in the company unit (*company _id*). What this allocation controls is that only customer datasets that belong to this company are visible. Furthermore, the employee is equipped with access rights (*permission _id*) (see the above description). These access rights control what information the employee is offered by the system. Thus, for example, an account manager (permission _id=3) cannot view any statistics above the

company itself (number of contracts, number of employees, contracts per employee, etc.). In order to enable employees to be represented, a reference to another employee is also maintained. As soon as an employee logs off and defines a representative, all information are also made available to the representative.

[00154] For authentication, every employee is assigned a user name as well as a password. Upon logon to the system, the user is requested to enter the company name as well as user name and the password. The server application checks based on these parameters whether there is a user belonging to the indicated company and subsequently verifies the password. When these unambiguous parameters agree, the employee can be identified. A session key is generated in the data bank that unambiguously identifies the logged on user.

BENEFICIARIES

[00155]

```
DROP        TABLE dapt_beneficiary _mode;
CREATE              TABLE dapt _beneficiary _mode
(
        id          INT4  PRIMARY KEY,
        remark            TEXT  NOT NULL

);

INSERT INTO dapt_beneficiary_mode (id,remark) VALUES (0, 'SINGLE'     );
INSERT INTO dapt_beneficiary_mode (id,remark) VALUES (1, 'PAIR'          );
INSERT INTO dapt_beneficiary_mode (id,remark) VALUES (2, 'COMPANY'  );

DROP            SEQUENCE        seq_dapt_beneficiary;
DROP            TABLE           dapt_beneficiary;
CREATE          TABLE           dapt_beneficiary
(

        id              INT4  PRIMARY KEY,

        mode_id         INT4  REFERENCES dapt_customer_mode(id)DEFAULT(0),

        firstName      TEXT  NOT NULL,
        lastName       TEXT  NOT NULL,
        title          TEXT,
        gender_id      INT4  REFERENCES dapt_gender(ID) NOT NULL,
        dateOfBirth    DATE  NOT NULL,

        firstName2     TEXT ,
        lastName2      TEXT ,
        title2         TEXT ,
        gender_id2      INT4   REFERENCES dapt_gender(ID) DEFAULT(0),
        dateOfBirth2    DATE DEFAULT CURRENT_DATE,

        accountNumber INT4 NOT    NULL,
        bankIdentificationCode    NOT NULL,
        bankDetail                TEXT  NOT NULL,
        company_id                INT4   REFERENCES dapt_company(id) NOT NULL
);

CREATE              SEQUENCE          seq_dapt_beneficiary;
```

[00156] Payouts of investments can generally be paid out to different accounts (of the beneficiary). It is thus not only a customer but also a beneficiary that is assigned to each contract.

CUSTOMER UNIT

[00157]

```
DROPTABLE dapt_customer_status;
CREATE      TABLE dapt_customer_status
(

        id                      INT4    PRIMARY KEY,
        remark                  TEXT  NOT NULL
);

INSERT INTO dapt_customer_status (id,remark) VALUES (0, 'REGISTERED'
                );
INSERT INTO dapt_customer_status (id,remark) VALUES (1, 'CONFIRMED'
        );
INSERT INTO dapt_customer_status (id,remark) VALUES (2, 'CHECKOUT'
                );
INSERT INTO dapt_customer_status (id,remark) VALUES (3,
'CHECKOUTCONFIRMED'         );
INSERT INTO dapt_customer_status (id,remark) VALUES (4, 'CHANGED');


DROPTABLE dapt_customer_mode;
CREATE                 TABLE dapt_customer_mode
(
        id              INT4  PRIMARY KEY,
        remark          TEXT  NOT NULL
);

INSERT INTO dapt_customer_status (id,remark) VALUES (0,'SINGLE'    );
INSERT INTO dapt_customer_status (id,remark) VALUES (1, 'PAIR' );
INSERT INTO dapt_customer_status (id,remark) VALUES (2,'COMPANY );

DROP          SEQUENCE          seq_dapt_customer;
DROP      TABLE          dapt__customer;
CREATE    TABLE          dapt__customer;
(

        id              INT4    PRIMARY KEY,

        mode_id         INT4    REFERENCES dapt__customer__mode(id) DEFAULT(0),

        firstName       TEXT    NOT NULL,
        lastName        TEXT    NOT NULL,
        title           TEXT    ,
        gender_id       INT4    REFERENCES dapt__gender(ID) NOT NULL,
        dateOfBirth     DATE    NOT NULL,

        firstName2      TEXT    ,
        lastName2       TEXT    ,
        title2          TEXT    ,
        gender_id2      INT4    REFERENCES dapt_gender(ID) DEFAULT(0),
        dateOfBirth2    DATE    DEFAULT CURRENT_DATE,

        company__id     INT4    REFERENCES dapt__company(id) NOT NULL,
        team__id        INT4    REFERENCES dapt_team(id)     NOT NULL,
        status_id       INT4    REFERENCES dapt__customer__status(id)   NOT
NULL
        street1         TEXT    NOT NULL,
        street2                 TEXT   ,
        city            TEXT    ,
        state           TEXT    ,
        zip             INT4    NOT NULL,
        country                 TEXT   NOT NULL,
        phone           TEXT    ,
        fax             TEXT    ,
        email           TEXT    ,
        www             TEXT    ,

        accountNumber INT4 DEFAULT(-1),
        bankIdent       INT4    DEFAULT(-1),
        bankDetail      TEXT    ,
        creator__id     INT4    REFERENCES dapt__employee(id) NOT   NULL,
        createdDate     DATE    NOT NULL,
        createdTime     TIME    NOT NULL,

        modifier_id     INT4    REFERENCES dapt__employee(id) NOT NULL,
        modifiedDate    DATE    NOT NULL,
        modifiedTime    TIME    NOT NULL
);

CREATE              SEQUENCE   seq__dapt__customer;
```

UNIT FOR STORING FUND COMPANIES

[00158]

```
DROP         SEQUENCE   seq__dapt__company__address;
DROP         TABLE dapt__company__address;
CREATE       TABLE         dapt__company__address

        id              INT4    PRIMARY KEY,
        company__id     INT4    REFERENCES dapt__company(ID) NOT NULL,
        name            TEXT    NOT NULL
        street1         TEXT    NOT NULL,

        street2         TEXT,
        city            TEXT    NOT NULL
        state           TEXT,
        zip                     INT4 NOT NULL,
        country                 TEXT   NOT NULL,
        phone           TEXT    NOT NULL,
        fax                     TEXT,
        email           TEXT,
        www                     TEXT,
        remark          TEXT,
        creator_id      INT4,   REFERENCES dapt__employee(id) NOT      NULL,
        createdDate     DATE    NOT NULL,
        createdTime     TIME    NOT NULL,

        modifier__id    INT4    REFERENCES dapt__employee(id) NOT NULL,
        modifiedDate    DATE    NOT NULL,
        modifiedTime    TIME    NOT NULL

);

CREATE             SEQUENCE seq__dapt__company__address;
```

DEPOSIT UNIT

[00159]

```
DROP SEQUENCE  seq__dapt__deposit;
DROP TABLE     dapt__deposit;
CREATE TABLE   dapt__deposit
(
        id                          INT4 PRIMARY KEY,
        customer__id                INT4 REFERENCES dapt__customer(id) NOT
NULL,
        name                        TEXT NOT NULL,
        deposit AccountNumber       INT4   NOT NULL,
        bankIdentificationCode             INT4 NOT NULL,
        bankDetail                  TEXT NOT NULL,
        UNIQUE (customer__id,depositAccountNumber, bankIdentification Code)

);

CREATE      SEQUENCE seq__dapt__deposit;
```

[00160] Arbitrary deposits can be assigned to each customer. The deposits are usually requested by the customer himself in writing in parallel (apart from the system). These written documents are available to the consultant. Securities are booked in this deposit.

FEE UNIT

[00161]

```
DROP SEQUENCE   seq_dapt_fee;
DROP TABLE        dapt_fee;
CREATE      TABLE dapt_fee
(

        id              INT4   PRIMARY KEY
        company_id      INT4   REFERENCES dapt_company(id) NOT NULL,
        name            TEXT   NOT NULL,
        perAnnum        INT4   NOT NULL,
        fee                    FLOAT NOT NULL,
        feeMinimum      FLOAT NOT NULL

        creator_id      INT4   REFERENCES dapt_employee(id) NOT NULL,
        createdDate     DATE   NOT NULL,
        createdTime     TIME   NOT NULL,

        modifier_id     INT4   REFERENCES dapt_employee(id),
        modifiedDate           DATE,
        modifiedtime           TIME,

        UNIQUE (name, fee, feeMinimum, perAnnum, company_id)

);
CREATE SEQUENCE seq_dapt_fee;
```

[00162] Fee models are allocated to each company. These fee models can contain minimum amounts and can also be marked as fees to be paid annually. Every securing contract is linked to a fee model.

CONTRACT UNIT

[00163]

```
DROPTABLE dapt_return_mode;
CREATE      TABLE dapt_return_mode
(

        id              INT4            PRIMARY KEY,
        remark          TEXT  NOT NULL
);

INSERT INTO dapt_return_mode(id,remark) VALUES (0,'ABSOLUTE');
INSERT INTO dapt_return_mode(id,remark) VALUES (1,'RELATIVE');

DROP    TABLE   dapt_contract_status;
CREATE      TABLE dapt_contract_status
(

        id              INT4   PRIMARY KEY,
        remark          TEXT  NOT NULL
);

INSERT INTO dapt_contract_status (id,remark) VALUES (0,'REGISTERED'      );
INSERT INTO dapt_contract_status (id,remark) VALUES (1,'CONFIRMED'       );
INSERT INTO dapt_contract_status (id,remark) VALUES (2,'ACTIVATED'       );
INSERT INTO dapt_contract_status (id,remark) VALUES (3,'MONITORED'       );
INSERT INTO dapt_contract_status (id,remark) VALUES (4,'TERMINATED'      );
INSERT INTO dapt_contract_status (id,remark) VALUES    (5,'CHECKOUT'     );
INSERT INTO dapt_contract_status (id,remark) VALUES (6,'CHECKOUTCONFIRMED) ;
DROP        SEQUENCE    seq_dapt_team;
DROP        TABLE       dapt_team;
CREATE TABLE            dapt_team
(

        id                              INT4   PRIMARY KEY,
        name            TEXT            DEFAULT('--'),
        advisor_id      INT4            REFERENCES dapt_employed(id) NOT NULL
        user_id         INT4            REFERENCES dapt_employed(id) NOT NULL
        UNIQUE (advisor _id, user_id)
);

CREATE SEQUENCE seq_dapt_team;

DROP    SEQUENCE seq_dapt_contract;
DROPTABLE dapt_contract;
CREATE TABLE   dapt_contract
(
id              INT4   PRIMARY KEY,
deposit_id      INT4   REFERENCES dapt_deposit(id) NOT NULL,
nsin                   INT4   NOT NULL
reference                     INT4   NOT NULL,

beneficiary_id         INT4          REFERENCES dapt_beneficiary(id) NOT NULL,
broker_id       INT4          INT4   REFERENCES dapt_team(id) NOT NULL,
company_id                    INT4   REFERENCES dapt_company(id) NOT NULL,
speculativePeriod             INT4

numberAssets                         FLOAT CHECK (numberAssets>=0),
purchaseDate                    DATE NOT NULL,
purchaseTime                    TIME  NOT NULL,
purchasePrice                   FLOAT NOT NULL CHECK (price>=0),

activation              FLOAT,
activationMode_id       INT4   REFERENCES dapt_return_mode,
activated               INT4
activationDate                  DATE NOT NULL,
activationTime          TIME   NOT NULL,

safeGuard               FLOAT NOT NULL,
safeGuardMode_id        INT4   REFERENCES dapt_return_mode(id),
safeGuardFee_id         INT4   REFERENCES dapt_fee(id) NOT NULL,

StoppLoss               FLOAT NOT NULL,
StoppLossMode_id        INT4   REFERENCES dapt_return_mode(id),
StoppLossFee_id                REFERENCES dapt_fee(id),

price                   FLOAT   NOT NULL,
priceDate               DATE    NOT NULL,
priceTime               TIME    NOT NULL,

threshold               FLOAT NOT NULL,
thresholdDate                   DATE NOT NULL,
thresholdTime                   TIME  NOT NULL,

status_id               INT4   REFERENCES dapt_contract_status
DEFAULT(0),
statusDate              DATE NOT NULL,
statusTime              TIME  NOT NULL,
statusEmployee_id       INT4   REFERENCES dapt_employee(id)NOT NULL,

systemReference         INT4   NOT NULL,
systemDate              DATE NOT NULL,
systemTime              TIME  NOT NULL,

volume                  FLOAT NOT NULL,

salePrice               FLOAT,
saleDate                DATE,
saleTime                TIME,

contractNumberINT4        DEFAULT (-1)

);

CREATE SEQUENCE seq_dapt_contract;
```

RATE DATA UNIT

[00164]

```
DROP SEQUENCE seq_dapt_quote;
DROP TABLE   dapt_quote;

CREATE TABLE   dapt_quote
(

        id              INT4 NOT NULL,
        nsin            INT4 PRIMARY KEY,
        reference       INT4 NOT NULL,
        name            TEXT NOT NULL,

        price           FLOAT NOT NULL CHECK (price>=0),
        date            DATE NOT NULL,
        time            TIME NOT NULL,

        UNIQUE (id, nsin, reference)

);

CREATE SEQUENCE seq_dapt_quote;
```

NOTIFICATION UNIT

[00165]

```
DROP        SEQUENCE seq_dapt_notification;
DROP        TABLE dapt_notification;

CREATE TABLE   dapt_notification
(

        id              INT4   PRIMARY KEY,
        target_id       INT4   REFERENCES dapt_permission(id) NOT NULL,
        company_id      INT4   REFERENCES dapt_company(id) NOT NULL,
        message         TEXT NOT NULL,
        link            TEXT NOT NULL,
        date            DATE NOT NULL,
        time            TIME NOT NULL
);

CREATE SEQUENCE seq_dapt_notification;
```

[00166] Account managers and consultants can themselves activate contracts or customer applications without having to wait for a confirmation of the company

administrator. The company administrator is notified of this bypassing of the actual work sequence with a message.

10074927 .021202